

# Theory of Computer Games, NTU Homework #1

Due date: 14:20 (UTC+8), October 26, 2017

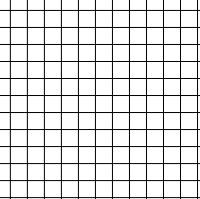
# Homework Description

- In this homework, you are asked to
  - Implement a solver of **Nonogram**
  - Compare the performance of **different** search algorithms

# Nonogram

				1					1					
				1	2	2		2	1					
	3	2	1	6	1	1	7	1	1	6	1	2	3	
5	5	9	11	2	3	3	1	3	3	2	11	9	5	5
5	3	2	1	1	3	3	3	3	3	1	1	2	3	5

			5	5
		3	5	3
		2	9	2
1	2	1	2	1
		1	11	1
		4	1	4
		4	1	4
				13
			6	6
				13
	1	2	2	1
		1	11	1
		2	9	2
		3	5	3
			5	5



				1					1			
				1	2	2		2	2	1		
	3	2	1	6	1	1	7	1	1	6	1	2
5	5	9	11	2	3	3	1	3	3	2	11	9
5	3	2	1	1	3	3	3	3	3	1	1	2

			5	5
		3	5	3
		2	9	2
1	2	1	2	1
		1	11	1
		4	1	4
		4	1	4
				13
			6	6
				13
	1	2	2	1
		1	11	1
		2	9	2
		3	5	3
			5	5

A 20x20 grid of black and white squares forming a pixelated image of a cat's face. The cat is facing forward, with its ears pointed upwards and its whiskers extending outwards. The image is composed of 400 individual squares, each either black or white, creating a high-contrast, pixelated effect.

- For each column and row, there is a **hint**
- For example, 5 5 means
  - There are 2 connected blocks strings
  - Each one has 5 connected blocks

# Random Problem Generator

- Random problem generator from TCGA 2016
  - `http://aigames.nctu.edu.tw/~hsuehch/nonogram/tcga2016/boardgen.py`
- Usage:
  - `./boardgen.py n num P1 P2 SEED`
  - `./boardgen.py 25 1000 0.5 0.35 12345`
    - n: size of board is n by n
    - num: number of test case
    - p1: max probability a cell is black
    - p2: min probability a cell is black
    - SEED: random seed

# Input

The figure displays two identical 10x10 grids illustrating the addition of two numbers. Each grid has a top row of digits from 0 to 9. Below this, there are four rows of numbers representing the addition process.

**Left Grid:**

- Row 1: 0 1 2 3 4 5 6 7 8 9
- Row 2: 12 12 12 12 12 12 12 12 12 12
- Row 3: 24 24 24 24 24 24 24 24 24 24
- Row 4: 24 24 24 24 24 24 24 24 24 24

**Right Grid:**

- Row 1: 0 1 2 3 4 5 6 7 8 9
- Row 2: 12 12 12 12 12 12 12 12 12 12
- Row 3: 24 24 24 24 24 24 24 24 24 24
- Row 4: 24 24 24 24 24 24 24 24 24 24

- Read from standard input.

\$1 // Problem Number

5 5 // Hint of first column, from  
up to down

- 
- 
- 

5 5 // Hint of last column

5 5 // Hint of first row, from left to right

- 
- 
- 

5 5 // Hint of last row

\$2 // Problem Number

•

•

•

```
$1
1 1 1 1 1 0 0 0 0 0 1 1 1 1 1
1 1 1 0 0 1 1 1 1 1 0 0 1 1 1
:
1 1 1 0 0 1 1 1 1 1 0 0 1 1 1
1 1 1 1 1 0 0 0 0 0 1 1 1 1 1
$2
:
```

- Write to standard output.
- For each case, output \$(Problem Number) in the first line. Then output  $n$  lines, each line contains  $n$  elements separated by space or tab.
  - 1: block
  - 0: non-block
- For example, see output file of boardgen.py.

# Standard Test Board

- Your program should at least pass the following test data
  - boardgen.py 5 10 0.5 0.3 514514999
  - boardgen.py 10 10 0.5 0.3 514514999
  - boardgen.py 15 10 0.5 0.3 514514999
  - The time limit is 60 seconds for each input file.
- Notification
  - The input file do not indicate the board size
  - You can caculate it from the number of lines of the input file.

# Solution Package

- Submit page: <http://w.csie.org/~tcg/2017/>
- Package structure:
  - Your ID [R05xxxxxx/B02xxxxxx/...]
    - **code** // A folder contains all your codes
    - **report.pdf** // Your report
- Compress your folder into a “**zip**” file
- Due to server limitation, the file size is restricted to **2M** bytes
- If your program has a pattern database and the size is greater than 2Mb, you can simply upload the code that generates the pattern database.



# What Should Be Included in Your Reprot?

- About your code
  - You can use any programming language.
  - How to **compile** and **run** your program.
  - Must be compiled and run under linux or windows 10.
  - If TA has difficulty in compiling your code, he may ask you to demonstrate the process.
  - What algorithm and heuristic you implement
- Experiment
  - The comparison between **different algorithms**.
  - Must include running time.
- Discussion
  - The game complexity analysis
  - The factors affect the performance of each algorithm
  - The factors affect the difficulty of Nonogram
  - Other observation or discussion

- Nonogram's wikipedia page
  - <https://zh.wikipedia.org/wiki/Nonogram>
- TCGA 2016 Nonogram Tournament
  - <http://aigames.nctu.edu.tw/~hsuehch/nonogram/tcga2016/>
- An on-line nonogram playing site
  - <http://www.puzzle-nonograms.com/>